

Pipelined CPU

Gwangmu Lee

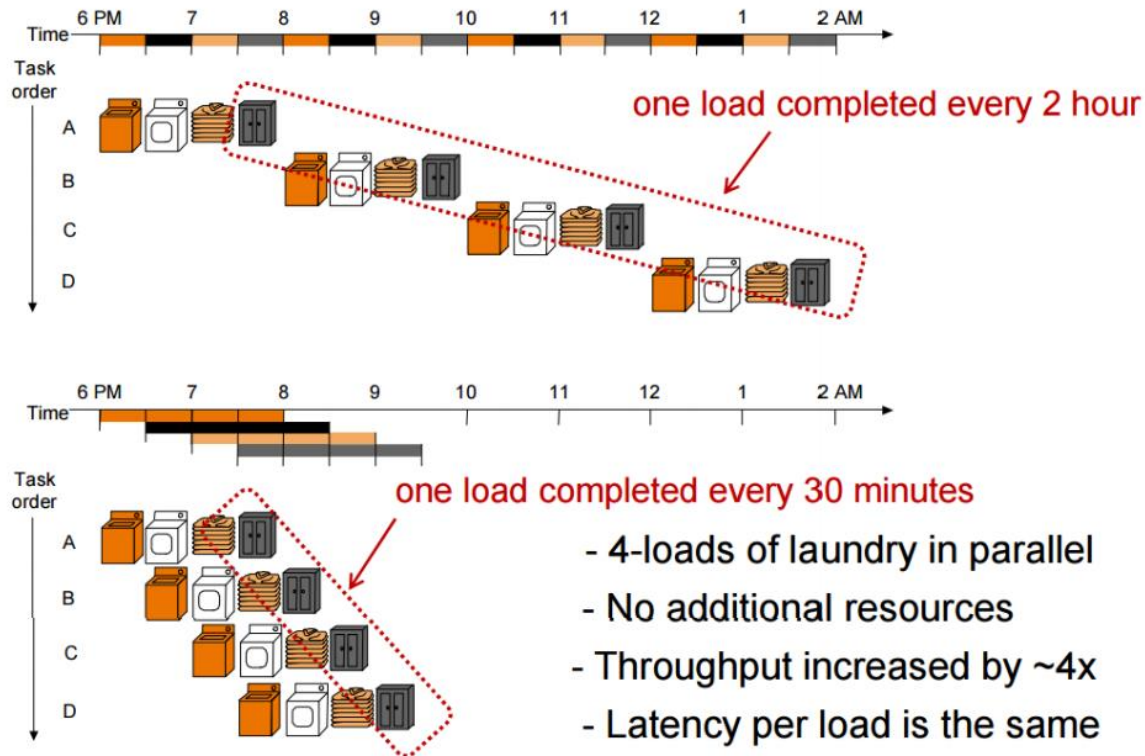
April 29, 2017

gwangmu@snu.ac.kr

Objectives

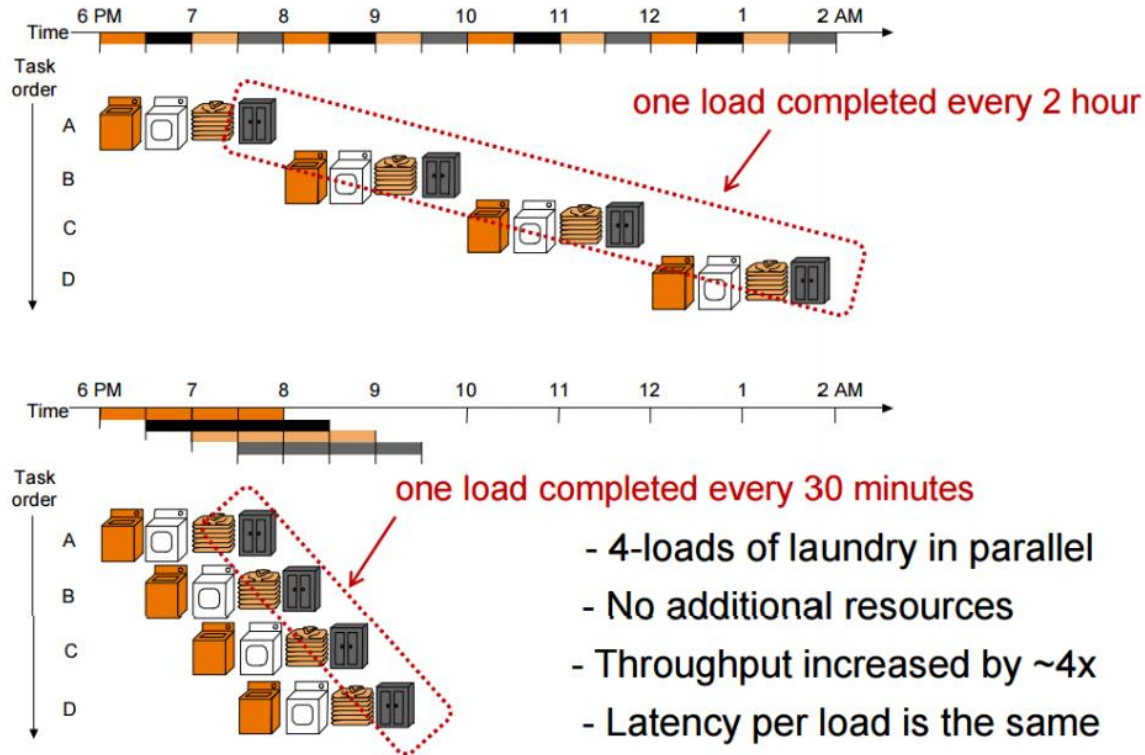
- Understand why pipelining improves the performance of a CPU.
- Understand what the control/data hazards are and how to resolve them.
- Design and implement a pipelined CPU.
 - Refer to the textbook (4.6 ~ 4.8) and lecture slides.

Why Pipelined? – Analogy



- Improve throughput with little additional HW.
 - **Task = A basket of laundry, Resource = Machines/Furniture**
 - Notice that a resource (e.g., a washing machine) can do a different task while other resources (e.g., a dryer) doing other tasks, simultaneously.

Why Pipelined? – Analogy



Task
Resource

Laundry

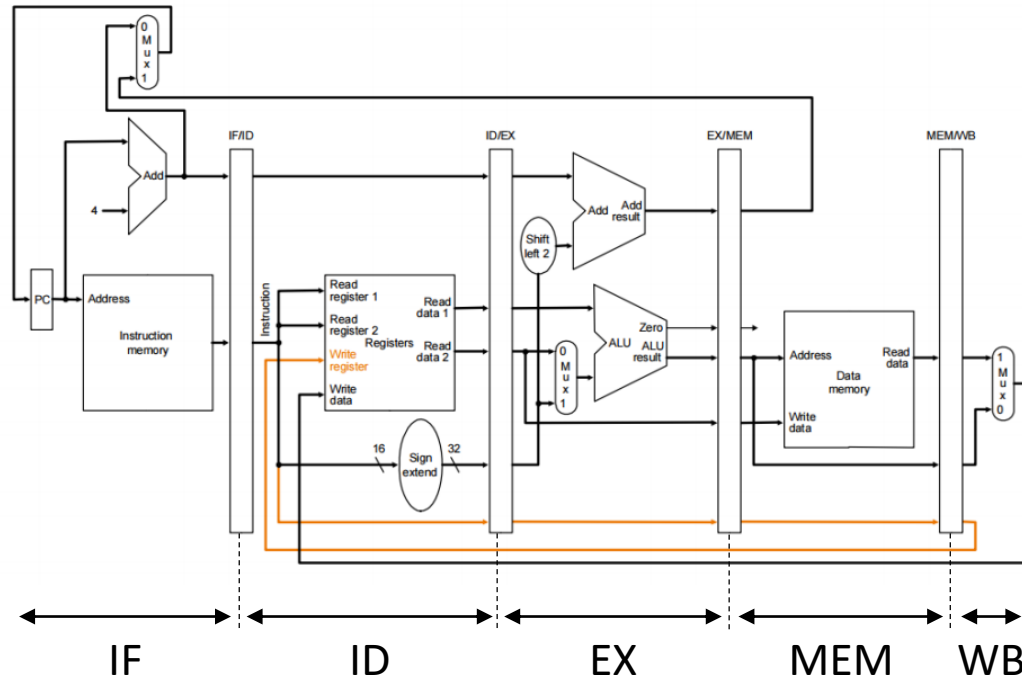
A basket of laundry
Machines/Furniture



CPU

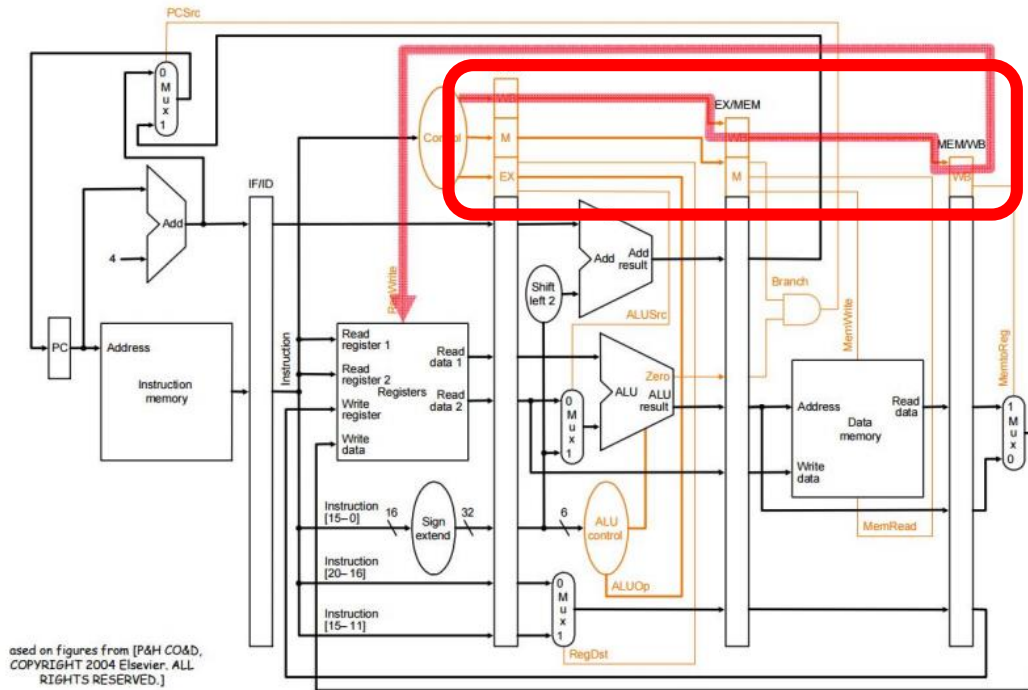
Instruction
Pipeline Stage (e.g., IF)

Pipelining – Datapath-only



- The intermediate data of an instruction flow through pipeline stages and pipeline latches.
 - IF → (IF/ID latch) –[next cycle]→ ID → (ID/EX latch) –[next cycle]→ ...

Pipelining – With Control



- Because each pipeline stage operates on different instructions, each stage should refer to a different set of control signals.
- Thus, control signals also flow through pipeline latch.
 - See the red box in the figure.

Hazards


- What is *pipeline hazards*?
 - The situations that prevent the next instruction from being executed during certain clock cycles.
- Common types of *hazards*
 - Data hazard
 - Control hazard
 - Structural hazard

Let's see what they are!

Hazards – Data hazard

	R/I-Type	LW	SW	Br	J	Jr
IF						
ID	read RF	read RF	read RF	read RF		read RF
EX						
MEM						
WB	write RF	write RF				

- RAW data hazard
 - Let's assume
 - I(A) is an **R/I-type or Load(LW)** instruction, and I(B) is an **R/I-type, Load(LW), Store(SW), Branch(BR), or Jr** instruction.
 - I(B) reads a register written by I(A).
 - I(B) cannot see the value written by I(A), if $\text{dist}(I(A), I(B)) \leq \text{dist}(ID, WB) = 3$.
($\text{dist}(x, y) :=$ pipeline distance btw x and y.)
 - *This is RAW data hazard.*

```
...  
I(A): LDR $1, 0x1000  
I(B): ADD $3,  $1, $2  
...
```


Hazards – Data hazard

- How to resolve?
 - Solution 1: Pipeline stall (“Let’s insert bubbles”).
 - Stop advancing PC and Disable latching.
 - Solution 2: Data forwarding
 - **Extra credit** if you implement this for this assignment!
 - c.f.) Data use/produce table for data forwarding.

	R/I-Type	LW	SW	Br	J	Jr
IF						
ID						use
EX	use produce	use	use	use		
MEM		produce	(use)			
WB						

Hazards – Control hazard

	R/I-Type	LW	SW	Br	J	Jr
IF	use	use	use	use	use	use
ID	produce	produce	produce		produce	produce
EX				produce		
MEM						
WB						

- Control hazard distance is at least 1.
 - Reason: IF can know whether (1) the current PC is branch or not, and if it is a branch, (2) the branch is taken or not, only after the current PC is decoded (ID).
- How to resolve?
 - Solution 1: Pipeline stall
 - Solution 2: Prediction (+ flush) **[Extra credit]**
 - Always taken, Always not taken, or other advanced prediction techniques.

Hazards – Structural hazard

- *Wikipedia is our friend. He explains all..*

Structural hazards

A structural hazard occurs when a part of the processor's hardware is needed by two or more instructions at the same time. A canonical example is a single memory unit that is accessed both in the fetch stage where an instruction is retrieved from memory, and the memory stage where data is written and/or read from memory.[3] They can often be resolved by separating the component into orthogonal units (such as separate caches) or bubbling the pipeline.

- Extracted from Wikipedia, "Hazard (computer science)".

- You don't need to bother with this in this assignment, because..
 - The memory has already two ports, one for IF and another for MEM.
 - Since our CPU is not a superscalar CPU, one execution unit suffices to execute one instruction per cycles.

Assignment Summary

- Implement a pipelined CPU.
 - Refer to the designs in the textbook (4.6 ~ 4.8) and lecture slides.
 - Your CPU module must pass all provided test cases.
- Compare the performance (multi-cycle CPU vs. pipelined CPU).
 - Your report must include the comparison on # of clock cycles to execute all test cases.

Good luck, everyone!

